Variational Dropout Sparsifies Deep Neural Networks

Dmitry Molchanov
dmitry.molchanov@skolkovotech.ru

Arseniy Ashuha
ars.ashuha@phystech.edu

Dmitry Vetrov
dvetrov@hse.ru

p(B|A)yesgroup.ru
Skoltech
Yandex
MIPT

Key results

Sparse Variational Dropout: a New Method for DNN Sparsification

- Compression of DNNs through sparsification of weight matrices
  - CIFAR-10 VGG: up to 70x compression
  - MNIST LeNet-5: up to 280x compression
  - No accuracy loss!
- Simple and easy-to-use:
  - Additive Gaussian noise on activations and a regularization term
  - Optimize w.r.t. **both** weights **and** noise variances ($\approx$ dropout rates)

Dropout and Stochastic Variational Inference

Dropout training optimizes the cross-entropy loss under stochastic setting:

$$-\mathbb{E}_\varepsilon \log p(Y \mid X, \widetilde{W} = W \odot \varepsilon) \to \min_W \qquad \varepsilon \sim Bernoulli(p)$$

Gaussian Dropout is similar but puts Gaussian noise on the weights:

$$\widetilde{W} = W \odot \varepsilon \qquad \varepsilon_{ij} \sim \mathcal{N}(1, \alpha) \qquad q(\tilde{w}_{ij}) = \mathcal{N}(w_{ij}, \alpha w_{ij}^2)$$

Noise over w_{ij} means that $\tilde{w}_{ij}$ is a random variable with distribution $q(\tilde{w}_{ij})$

Stochastic Variational Inference:

$$\underbrace{-\mathbb{E}_{q(\widetilde{W} \mid W, \alpha)} \log p(Y \mid X, \widetilde{W})}_{\text{Data-term (e.g. cross-entropy loss)}} + \underbrace{\mathrm{D_{KL}}(q(\widetilde{W} \mid W, \alpha) \,\|\, p_{prior}(\widetilde{W}))}_{\text{Regularizer}} \to \min_{W, \alpha}$$

- The true posterior distribution over weights $\widetilde{W}$ is approximated by q

$$\mathrm{D_{KL}}(q(\widetilde{W} \mid W, \alpha) \,\|\, p(W \mid X, Y)) \to \min_{W, \alpha}$$

- Just a slightly different loss function; implementation is basically the same

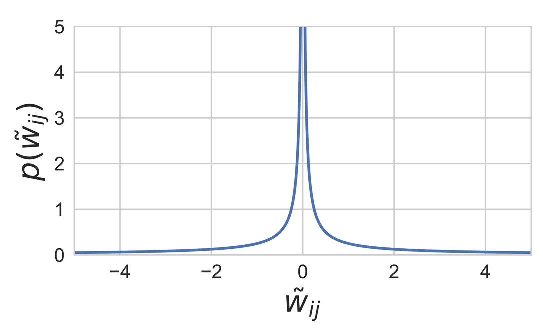
Sparse Variational Dropout

- Gaussian Dropout posterior distribution

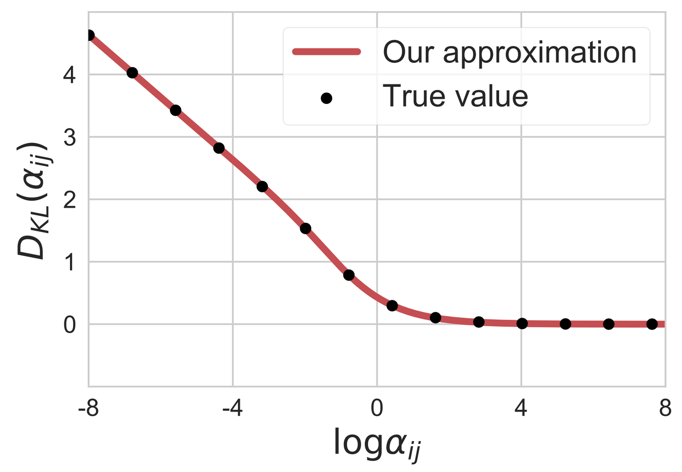
$$\widetilde{W} = W \odot \varepsilon \quad \Leftrightarrow \quad \tilde{w}_{ij} \sim \mathcal{N}(w_{ij}, \alpha_{ij} w_{ij}^2) \quad = \quad q(\tilde{w}_{ij} \mid w_{ij}, \alpha_{ij})$$

- Sparsity-inducing log-uniform prior [2] **favors large dropout rates**

$$p(\tilde{w}_{ij}) \propto \frac{1}{|\tilde{w}_{ij}|}$$



- Now we can optimize w.r.t **both** weights w_{ij} **and** dropout rates α_{ij}
- The KL divergence only depends on dropout rates α_{ij}
- The KL divergence is intractable, but can be accurately approximated



Approximation of the KL divergence:

Black the true value
Red our approximation

Intuition for Sparsity

The regularizer favors large dropout rates α_{ij}

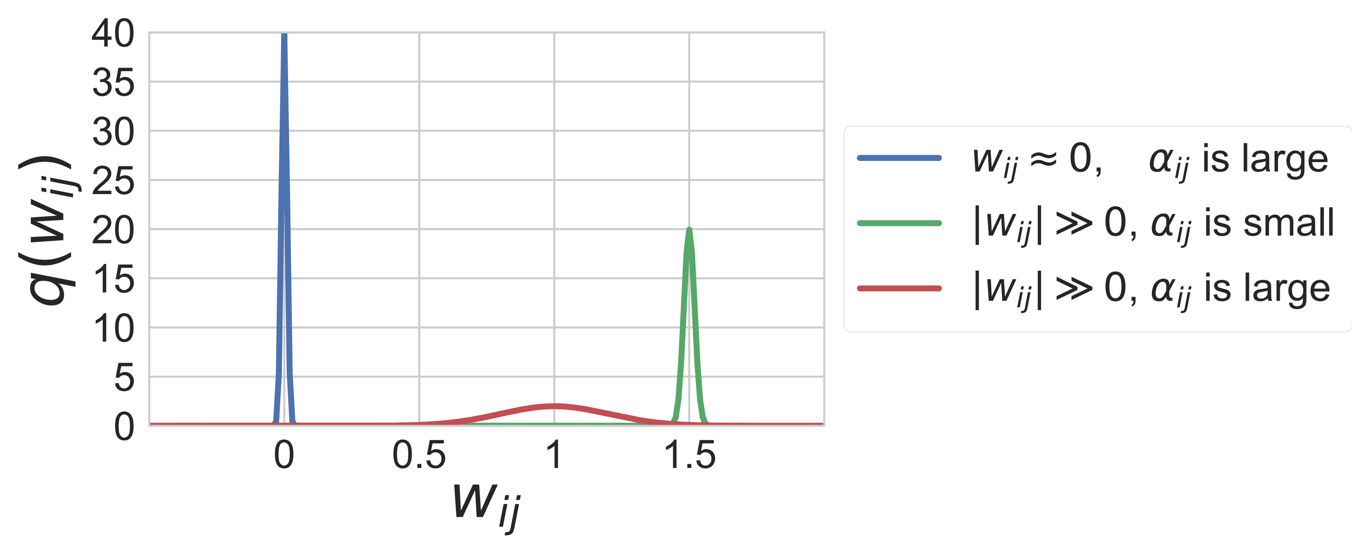
$$\mathrm{D_{KL}}(q(\tilde{w}_{ij} \mid w_{ij}, \alpha_{ij}) \,\|\, p(\tilde{w}_{ij})) \text{ decreases when } \alpha_{ij} \to \infty$$

Infinitely large dropout rate ($\alpha_{ij} \to \infty$) means:

- Infinitely large noise over the weight corrupts the data-term if $w_{ij} \neq 0$

$$\tilde{w}_{ij} = w_{ij} \cdot (1 + \sqrt{\alpha_{ij}} \cdot \varepsilon_{ij})\big|_{\alpha_{ij} \to +\infty}$$

- Equivalent binary dropout rate $p = \frac{\alpha}{1+\alpha} \to 1$, so $\tilde{w}_{ij} = 0$ during training
- Data-term controls the accuracy and prohibits to set all α's to infinity

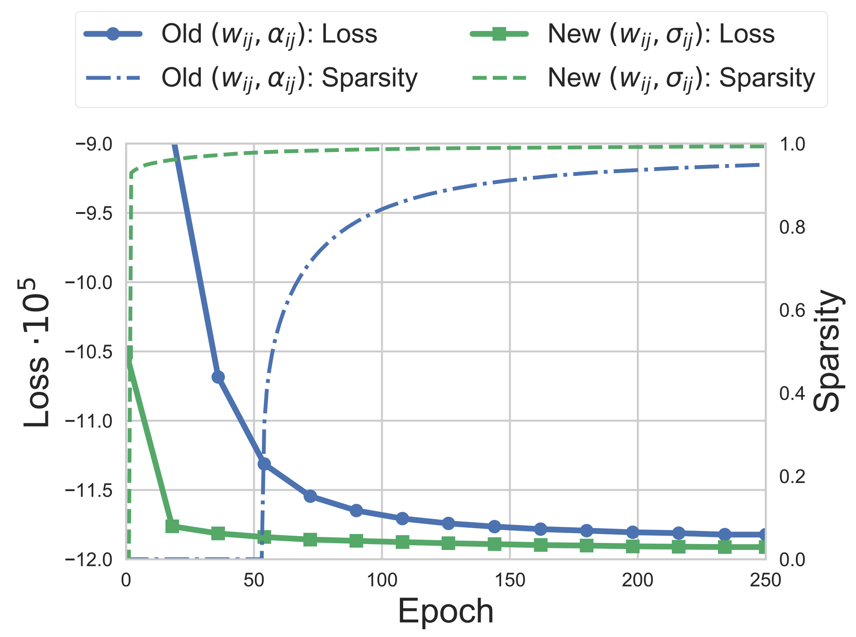


Sparse Variational Dropout Training

Variance Reduction 1: Additive Noise Parameterization

Optimize the loss $\mathcal{L}$ w.r.t. w_{ij} and $\sigma_{ij}^2 = \alpha_{ij} w_{ij}^2$

$$\frac{\partial \mathcal{L}}{\partial w_{ij}} = \frac{\partial \mathcal{L}}{\partial \tilde{w}_{ij}} \cdot \frac{\partial \tilde{w}_{ij}}{\partial w_{ij}}$$



Before: $\tilde{w}_{ij} = w_{ij}(1 + \sqrt{\alpha_{ij}}\varepsilon_{ij})$ $\quad \frac{\partial \tilde{w}_{ij}}{\partial w_{ij}} = 1 + \sqrt{\alpha_{ij}}\varepsilon_{ij}$ $\leftarrow$ **noisy!**

After: $\tilde{w}_{ij} = w_{ij} + \sigma_{ij}\varepsilon_{ij}$ $\quad \frac{\partial \tilde{w}_{ij}}{\partial w_{ij}} = 1$ $\leftarrow$ **no noise!**

Variance Reduction 2: Sample activations instead of weights (LRT [1, 2])

Before: $B = A\widetilde{W}$ $\quad \widetilde{W} = W + \sigma \odot \varepsilon$ $\quad \varepsilon_{ij} \sim \mathcal{N}(0, 1)$

After: $B \sim q(B)$ $\quad B = AW + \sqrt{A^2\sigma^2} \odot \varepsilon$ $\quad \varepsilon_{ij} \sim \mathcal{N}(0, 1)$

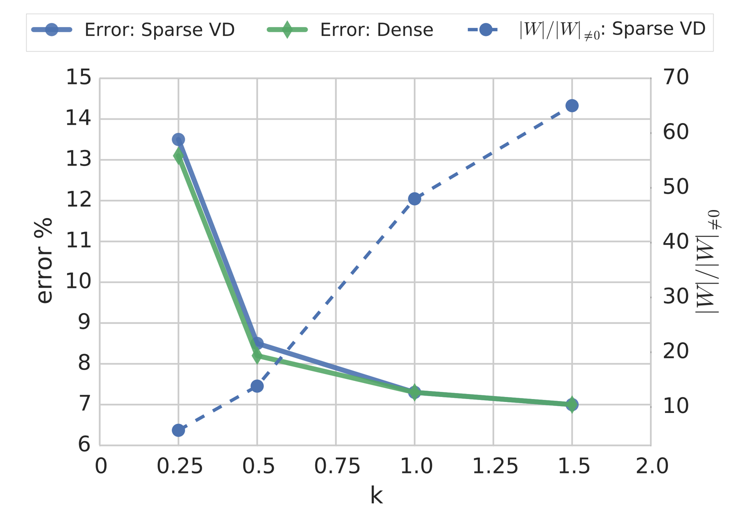
(all $\sqrt{\cdot}$ and $\cdot^2$ operations are element-wise)

Experiments: MNIST LeNet-5

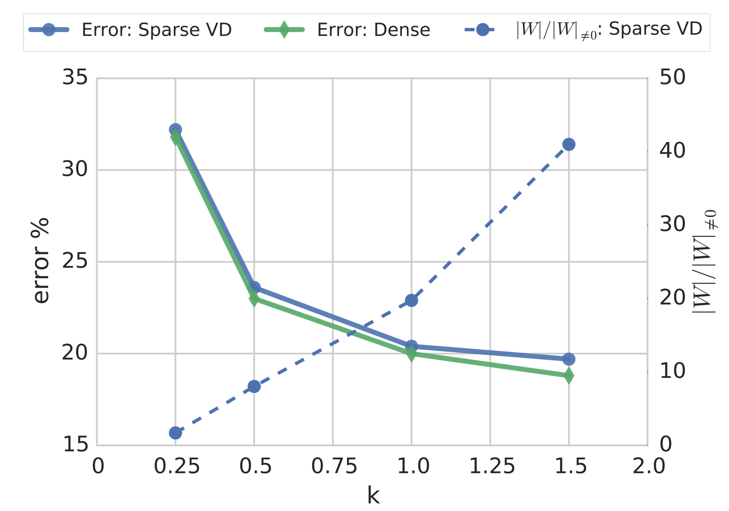
Method	Error %	Sparsity per Layer %	$\frac{\lvert W \rvert}{\lvert W_{\neq 0} \rvert}$
Original	0.80		1
Pruning [3]	0.77	$34 - 88 - 92.0 - 81$	12
DNS [4]	0.91	$86 - 97 - 99.3 - 96$	111
SWS [5]	0.97		200
Sparse VD	0.75	$67 - 98 - 99.8 - 95$	**280**

Comparison of different sparsity inducing techniques on the LeNet-5 architecture. Our method provides the highest level of sparsity with a similar accuracy.

Experiments: VGG-like on CIFAR-10 and CIFAR-100



(a) Results on the CIFAR-10 dataset



(b) Results on the CIFAR-100 dataset

Accuracy and sparsity level for VGG-like architectures of different sizes. The number of neurons and filters scales as k. Green: Binary Dropout. Blue: Sparse VD

Neuron-wise compression per layer for $k = 1.5$ (only whole neurons removed):

1x−1x−1x−1x−2x−3x−14x−9x−10x−85x−6x−8x−2x−2x−1x

The model is so sparse it even has neuron-wise sparsity!

Experiments: Random Labels [6]

Dataset	Architecture	Train Acc.	Test Acc.	Sparsity
MNIST	FC + BD	100%	10%	—
MNIST	FC + Sparse VD	10%	10%	100%
CIFAR-10	VGG + BD	100%	10%	—
CIFAR-10	VGG + Sparse VD	10%	10%	100%

Unlike Binary Dropout (BD), Sparse VD does not overfit on randomly labeled data and yields an empty network. It is an optimal architecture for this task!

Follow-up Papers

Bayesian Sparsification of Recurrent Neural Networks

arXiv link: goo.gl/uL94q1

- Up to 200x compression!
- Visit the poster on *"Bayesian Sparsification of RNNs"* at the *Learning to Generate Natural Language, Workshop, ICML'17*

Structured Bayesian Pruning via Log-N Multiplicative Noise

arXiv link: goo.gl/CzVBWP

- Compression and acceleration via group-wise sparsity
- Arbitrary pattern of structured sparsity

Links and References

Project Page: goo.gl/2D4tFW

1. Wang Sida and Christopher Manning. Fast dropout training, 2013
2. Diederik Kingma, Tim Salimans and Max Welling. Variational dropout and the local reparameterizationtrick, 2015
3. Song Han et al. Deep Compression: Compressing DNNs with Pruning, Trained Quantization and Huffman Coding, 2016
4. Yiwen Guo, Anbang Yao and Yurong Chen. Dynamic Network Surgery for Efficient DNNs, 2016
5. Karen Ullrich, Edward Meeds and Max Welling. Soft Weight-Sharing for Neural Network Compression, 2017
6. Chiyuan Zhang et al. Understanding deep learning requires rethinking generalization, 2017